

# Codice maligno generico e finalizzato

## Lezione 10 di Sicurezza dei sistemi informatici 1

Docente: Giuseppe Scollo

Università di Catania, sede di Comiso (RG)

Facoltà di Scienze Matematiche, Fisiche e Naturali

Corso di Studi in Informatica applicata, AA 2007-8

### Indice

1. Codice maligno generico e finalizzato
2. virus FAQ
3. il virus Brain
4. il worm di Morris
5. Code Red
6. Web bug
7. trapdoor
8. attacchi granulari
9. canali nascosti

## virus FAQ

i virus infettano solo i sistemi operativi Microsoft?

**no:** nessun sistema operativo è immune ad attacchi da virus

i virus possono modificare i file nascosti e/o di sola lettura?

**sì:** tali protezioni sono stabilite dal software, dunque modificabili da codice maligno

i virus possono essere incorporati solo in alcuni tipi di file (programmi, dati, documenti)?

**no:** qualsiasi tipo di file può contenere codice maligno; perché questo entri in azione occorre che se ne lanci l'esecuzione, ma ciò può essere effettuato dal programma di apertura del file

i virus si diffondono solo sui dischi o per posta elettronica?

**no:** qualsiasi mezzo di condivisione dell'informazione può essere un tramite della diffusione

i virus possono restare in memoria dopo un riavvio della macchina?

**no:** la RAM è volatile; però un virus che si installi su memoria di massa, ad esempio nel settore di avvio di un disco, può resistere al riavvio e quindi ripresentarsi

i virus possono infettare l'hardware?

**no:** oggetto di un'infezione virale è sempre l'integrità dell'informazione memorizzata su qualche supporto fisico, mai l'integrità fisica del supporto stesso

tutti i virus contengono codice maligno?

**no:** alcuni sono innocui; inoltre, la tipica diffusione virale può anche avere applicazioni utili, ad es. la compressione di file (con certe caratteristiche) in un supporto per economia di spazio

## il virus Brain

uno dei primi, e più noti, **virus del settore di avvio** (1986)

deve il nome alla sua caratteristica di modificare in "**(c) Brain**" l'etichetta del dischetto infetto

innocuo nella versione originale, ma ne sono state realizzate diverse varianti, fra cui alcune maligne

si è "estinto" per l'obsolescenza del floppy disk

**come funziona** (in breve):

il virus sovrascrive il settore di avvio ed altri sei settori del disco attaccato; di questi: uno contiene una copia del codice di avvio originariamente posto nel settore di avvio, due contengono il codice del virus, gli altri tre sono una copia dei primi tre  
il virus si appropria dell'**interrupt di lettura** del disco, riscrivendo la tabella degli indirizzi di gestione degli interrupt, e assegna all'interrupt 6 (inutilizzato) l'indirizzo della precedente routine di gestione dell'interrupt di lettura del disco  
quindi il virus gestisce gli interrupt di lettura e li analizza:  
se l'interrupt non riguarda il settore di avvio, genera un interrupt 6 (gestione originaria)  
altrimenti restituisce il contenuto del settore di avvio originale, da uno dei settori sovrascritti

**cosa ha permesso di imparare:**

prototipo di impiego di tecniche successivamente divenute standard nella realizzazione di virus: occultamento nel settore di avvio, intercettazione e analisi delle interruzioni

**per saperne di più:** [http://en.wikipedia.org/wiki/\(c\)Brain](http://en.wikipedia.org/wiki/(c)Brain)

## il worm di Morris

un caso singolare di worm dannoso per un errore del suo ideatore: Robert T. Morris Jr. (1988)

innocuo nelle sue intenzioni, il worm era programmato per negoziare la sua replicazione in un host attaccato, così da impedire la proliferazione di copie in sistemi già colpiti

un difetto in questo meccanismo ha avuto per conseguenza proprio la proliferazione indesiderata, che causava rapidamente il degrado delle prestazioni della macchina colpita

**obiettivi** del worm :

**individuare** le potenziali vittime dell'attacco: sistemi Unix non ancora infetti

**diffondere** l'infezione

**occultare** la propria presenza

**come funziona** (in breve): sfrutta **tre** vulnerabilità dei sistemi Unix dell'epoca:

leggibilità pubblica del file con le password crittografate degli utenti:

la scelta non infrequente di password **"facili"** da indovinare ha permesso al

worm

di attivare la propria esecuzione mediante login remoto da sistemi già colpiti

**buffer overflow** nel programma di sistema fingerd

**trapdoor** nel programma di sistema sendmail (in modalità debug )

**cosa ha permesso di imparare:**

contromisure alle suddette vulnerabilità, ma soprattutto necessità di centri di informazione sulla sicurezza dei sistemi informatici, ad es. il **CERT**:

<http://www.cert.org>

## Code Red

il worm **più dannoso** nella storia di Internet sinora: risale all'estate 2001

il worm di Morris provocò l'arresto o la disconnessione di circa 6000 sistemi:

quelli colpiti da Code Red furono 750.000

impressionante la **rapidità di propagazione** dell'infezione:

il 19 luglio Code Red infettò più di 250.000 sistemi in appena 9 ore

**obiettivo** del worm :

infettare numerosi sistemi per sferrare un attacco distribuito alla disponibilità di

siti Web: Distributed Denial of Service (DDoS)

**come funziona** (in breve): sfrutta un buffer overflow nel software Microsoft IIS, per:

iniettare un cavallo di Troia nel Web server colpito, col quale

acquisire i privilegi necessari ad esplorare la rete (interrogando la porta 80 delle destinazioni) per individuare altri host che abbiano la stessa vulnerabilità, e

diffondere quindi l'infezione

in giorni e orari prefissati, sferrare l'attacco DDoS, sincronizzato con gli altri host infetti

la presenza di Code Red si faceva notare per una deliberata alterazione del sito Web attaccato

**cosa ha permesso di imparare:**

anche Code Red sembra avere avuto scopo essenzialmente dimostrativo (ma il danno dovuto al blocco dei servizi è stimato nell'ordine di 2 miliardi di dollari)

l'acquisizione di privilegi di Code Red avrebbe ben potuto dar luogo ad **altre minacce**

il worm ha mostrato alto grado di **polimorfismo**, e rapida diffusione grazie al **multithreading**

## Web bug

un **Web bug** consiste di codice HTML, incorporato in una pagina Web o in un messaggio di posta elettronica, finalizzato alla raccolta di informazioni sul visitatore del sito o destinatario del messaggio, **senza che questi ne sia consapevole**

quest'ultimo aspetto rende controversa la questione della legalità dei Web bug

**come funziona** (in breve):

il codice è "nascosto" nel TAG di un'immagine di dimensione minima, tale dunque da non essere percepita

la visita della pagina o l'apertura del messaggio (con interpretazione del codice HTML) causa l'invio di una richiesta HTTP ad un (altro) server, dove si raccolgono le informazioni desiderate

queste possono andare dalla semplice collezione di statistiche aggregate (anonime) alla raccolta di informazioni personali riservate, ad es. quando al servizio della richiesta il server associa un **cookie** precedentemente installato nel browser della "vittima", inconsapevole dello spionaggio del suo comportamento

**contromisure:**

rilevazione della presenza di Web bug :

[www.bugnosis.org](http://www.bugnosis.org) (solo per IE): il sito ha anche una ricca FAQ sull'argomento

[snort.org/pub-bin/sigs.cgi?sid=2925](http://snort.org/pub-bin/sigs.cgi?sid=2925) è un antidoto open source al problema

disabilitazione (eventualmente selettiva) dei cookie

## trapdoor

**definizione:** accesso non documentato alle funzionalità di un programma

possono avere sia finalità **benigne** che **maligne**

**perché** possono restare delle trapdoor nei programmi, dopo lo sviluppo:  
**dimenticate**

lasciate **intenzionalmente**, per:

**collaudo**

**manutenzione**

**attacchi**

**contromisure:** le trapdoor sono utili alle attività di sviluppo, collaudo e manutenzione del software, ma potenzialmente pericolose durante il suo esercizio, dunque ne andrebbe predisposto un **uso controllato**, che ad es.:

distingua le versioni di sviluppo da quelle di esercizio del software

imponga e verifichi la **chiusura** delle trapdoor nelle versioni di esercizio

## **attacchi granulari**

**vulnerabilità:** tolleranza dei programmi agli errori di troncamento o arrotondamento nell'esecuzione di operazioni aritmetiche

**inerente** ai limiti di precisione della rappresentazione di dati numerici

**profilo dell'attaccante:** programmatore che dirotta a proprio vantaggio **piccole frazioni** degli importi di **molte operazioni o transazioni finanziarie**

è essenziale al successo dell'attacco che l'entità della differenza fra importo giusto e importo "corretto" (a proprio vantaggio) dall'attaccante sia talmente piccola da passare inosservata

**persistenza** degli attacchi granulari:

questi sono tanto più efficaci quanto più lungo è il tempo necessario alla manifestazione di **effetti significativi**

sebbene le caratteristiche di un attacco granulare facciano facilmente passare inosservata ciascuna singola correzione indebita, il rapido accumulo di flussi finanziari su un'unica destinazione può indurre sospetti e quindi sollecitare controlli più accurati

## **canali nascosti**

**scopo:** accesso non autorizzato a informazioni riservate

**minaccia:** violazione della riservatezza

**mezzo frequente:** cavallo di Troia

**problema** (dell'attaccante): nascondere il canale

**soluzioni tecniche:**

variazioni poco appariscenti nel **formato di output** di dati non riservati  
canali di archiviazione, blocco di file, creazione temporanea di file grandi  
comunicazione **sincronizzata** fra cavallo di Troia e programma-spia

**canali a tempo**

risorsa condivisa: il tempo (CPU time)

**contromisure:**

analisi delle risorse condivise (a tempo di esecuzione)

analisi dei flussi di informazione, nel codice sorgente (analisi statica)

**efficacia delle contromisure:** limitata, le potenzialità di sviluppo di nuove tecniche di occultamento di canali informativi sono illimitate